TECHNICAL RESEARCH REPORT

On-Line Detection of Distributed Attacks from Space-Time Network Flow Patterns

by J.S. Baras, A.A. Cardenas, V.Ramezani

TR 2003-2

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2003		**-**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
On-Line Detection of Distributed Attacks from Space-Time Network Flow Patterns	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Army Research Office,PO Box 12211,Research Triangle Park,NC,27709**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**		**9**	

ON-LINE DETECTION OF DISTRIBUTED ATTACKS FROM SPACE-TIME NETWORK FLOW PATTERNS

J.S. Baras*, A.A. Cardenas, V. Ramezani
Electrical and Computer Engineering Department
and the Institute for Systems Research
University of Maryland
College Park, MD, 20742

ABSTRACT

Parametric and non-parametric change detection algorithms are applied to the problem of detecting changes in the direction of traffic flow. The directionality of the change in a network flow is assumed to have an objective or target. The particular problem of detecting distributed denial of service attacks from distributed observations is presented as a working framework. The performance of our change detection algorithms is evaluated via simulations.

1. INTRODUCTION

We are interested in detecting and classifying anomalous changes in the behavior of a network caused from distributed sources of the disturbance, including maliciously planned attacks with goal the disruption of the network. More specifically we are interested in detecting changes in the network flow, identifying abnormal changes, and extracting the characteristics of the changes, as soon as possible. Examples include the spreading of active worms through web servers, email viruses and distributed denial of service attacks.

Networks of interest include communication networks as well as sensor networks. We are principally interested in developing efficient and high performance algorithms for these problems while utilizing only passive monitoring of the time histories of network flows and other key network parameters at a small subset of nodes. In particular we consider a subset of highly connected nodes from where we can infer properties about the network state. Detecting a problem such as an attack or intrusion early in its development or spreading, before it has reached its full force can help in the assignment of resources to guarantee a reliable operation or in executing a rapid response to contain or nullify an attack. For example a denial of service attack becomes apparent in its final phase by observing the traffic flow in the network. We are interested in the "quickest detection" problem when the attack is distributed and coordinated from several nodes against a targeted node.

We investigate first the problem of a spreading congestion attack that incrementally compromises nodes. The behavior pattern as observed by different nodes in the network will be different from a panic mode (flash crowd). This problem is investigated as a step towards analyzing more complex distributed attacks.

A typical distributed denial of service attack involves sending a large number of packets from multiple sources to a single destination causing excessive amounts of endpoint, and possibly transit network bandwidth to be consumed. (Houle and Weaver, 2001). Our goal is to detect when a distributed denial of service is taking place in one sub-network of a transit (core) network comprised only on routers. We are assuming the transit network itself is not the target of the attack, but it is being used by the attack to reach the victim.

Various techniques have been proposed for mitigation of denial of service attacks that require the identification of the routers participating (involuntarily) in the attack. Most of these techniques consume a significant amount of router resources so it is advisable to use them only when needed. Some examples provided in some Cisco Routers are TCP Intercept and "Committed Access Rate." Some related work is also presented in (Bohacek, 2002) where the key step in the proposed denial of service mitigation algorithm consists in identifying the routers forwarding the malicious traffic.

We are addressing this monitoring and detection of abnormal behavior problem as a space-time inference problem. That is we look at the collection of the time histories at the sensing nodes together and not in isolation. This set of time histories constitutes our "monitoring data" or observations. We consider parametric and non-parametric models for the monitored statistics. The connectivity, logical and physical, of the nodes and the topology of the network influence the time history of the observations, and consequently the form of the algorithm and its properties. We provide a novel formulation of the problem as sequential space-time change detection on a graph. The mathematical techniques we use for detecting an attack are thus based on change detection theory. In a distributed environment a small change in local nodes can be correlated with the state at different nodes to provide a global view and early warning about the state of the

network. Distributed change detection problems on graphs are a novel formulation and problem.

More specifically for the denial of service attack problem, we use a "directionality" framework, which gives us a way to compute the severity and directionality of the change. The "severity" represents a composite hypothesis test that can be solved explicitly when the data are Gaussian. We also introduce a heuristic distributed change detection mechanism for "correlating" the alarms in a subset of monitored nodes. Given an alarm as a pair (direction and severity) we correlate the severity of the alarms with alarms from other nodes in the "same direction."

Finally we are investigating the effects of mobility on the solution to these problems. Mobility creates changing topology of the underlying network and of the associated graph. The resulting problems are treated in the framework of dynamic space-time inferencing based on dynamic graph models.

2. DETECTION OF SELF-PROPAGATING CODE

We have investigated the problem of detecting self-propagating code (worms) spreading over a network. Most active worms spreading over the Internet compromised hosts in a random way. For example Code Red spread by launching 99 threads that generated random IP addresses. The worm itself is normally small (Code Red I was about 4KB) and it only takes 40 bytes for a TCP SYN packet to determine if a service is accessible. So in order to detect self-propagating code we should add semantic information, which might vary depending on the worm (Nimda used several ways to spread itself). The most general-purpose information we might need is the number of TCP SYN packets seen. A distributed observation of a rise in TCP SYN packets (or a rise in the rate of change in the arrival of these packets) in several nodes of the network might indicate the presence of a worm scanning IP addresses for vulnerable hosts. The use of host unreachable messages and connection attempts to routers as a way of detecting worms will be less reliable while the worm is "getting off the ground" if it uses a hit-list scanning (Staniford et al. 2002.) The observations can be made at different participating ISPs enforcing policies for blocking self-propagating code once it is detected.

Our overall goal is to develop automated mechanisms for detecting worms based on their spread traffic patterns from widespread sensors (later attacks). In analyzing the associated segmentation of flows in complex networks, past research has revealed that the connectivity of the network influences the properties of spreading mechanisms for various types of attacks. In this context the behavior of different types of networks has been investigated (at least on a preliminary basis): random graphs, small-world networks (small path length, unusually large clustering coefficient), scale-free networks. For example it is now well known that the Internet router topology has a heavy tail distribution, which manifests in a core connected group and a statistically significant number of low degree nodes (a so called scale-free network) (Albert and Barabasi, 2002.)

Various classes of random graphs/networks have recently attracted attention in relation to various dynamic properties of the Internet and other networks. These include the Erdös-Rényi "random graph" model, where one starts with N nodes and connects every pair of nodes with probability p, ending up with a graph with approximately $pN(N-1)/2$ edges distributed randomly, and with the typical distance between two nodes scaling as $log(N)$. In such a network *clusters* of "like" or "close" nodes form. A key parameter in the analysis and discrimination between graphs is the *clustering coefficient* of a node *i.* Consider a node i with k_i edges connecting it to k_i other nodes. If nearest neighbors were part of the cluster there will be $k_i(k_i-1)/2$ edges between them. The clustering coefficient of the network at node i is the ratio between edges that actually exist between these k_i nodes, E_i, and the total number $C_i = 2E_i/(k_i(k_i-1))$. The network clustering coefficient is the average of the C_i.

It turns out that the clustering coefficients, and more importantly their variations, are very useful in classifying different types of networks. In a random graph $C = p$, while in real networks C is much larger than in a comparable random graph. Another important parameter in the description of a network is the *node degree*, i.e. the number of edges emanating from a node in a graph. The important discriminant is the *degree distribution*: Pr[a randomly selected node has exactly k edges] = P(k). In a random graph the majority of nodes have approximately the same degree close to the average degree of the network k_{av}, and the degree distribution is Poisson with peak P(k_{av}). On the other hand in most real large networks degree distribution deviates significantly from Poisson: it has a power law tail $P(k) \approx k^{-\gamma}$ (scale-free networks). Current research in dynamics of complex networks tries to identify basic dynamic interactions to explain these characteristics.

Recent studies have revealed on a preliminary basis, that graph topology, and in particular its classification from the perspectives of clustering coefficients and degree distribution, is intimately related to the robustness of the network when there is failure or attack. A key question is: "what happens to average path length between two nodes when the network fails (i.e. nodes are disabled at random) *vs* when the network is under attack (i.e. nodes with highest degree are disabled first). It turns

out that there are strong indications that scale free networks are very robust to random failures but susceptible to targeted attacks, while ad hoc networks are very robust to targeted attacks.

Figure 1 below, illustrates a typical scale-free network. As can easily be seen in such a model there is a set of subnetworks connected through a much smaller subset of key nodes (routers). In such a network the distribution of the node degree is heavy tailed, meaning that there is higher than typical chance that there are some nodes with many edges connected to them. Such nodes help spread attacks rapidly and are themselves primary targets of attacks. The network of Internet routers and the WWW are well known examples of such networks.

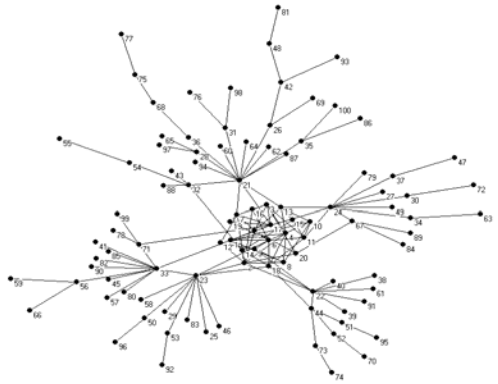

Figure1: The synthetic experimental network

The analysis of active worm spreading on graphs, provides a first step towards the analysis of spreading of more sophisticated attacks in a network. Fast spreading Internet worms (active worms) do not resemble traditional models of epidemic spreading on a graph. Traditional epidemic spreading on graphs considered nodes infecting only their neighbors. On the other hand in the Internet router graph the spread of active worms (Code Red, Nimda) is not limited to diffusion schemes due to the utilization of random scanning of nodes whose path length is in general greater than one. We have investigated experimentally the spreading of active worms using random scanning in a synthetic network. The time (number of hops) it takes to infect all the nodes is longer in regular lattices than in small world or random graphs. Naïve change detection would use a Poisson process for the beginning of connections and a sudden increase (in measured traffic rate) would create an alarm. Aggregation (fusion) of different sensors (located at different nodes) would prevent false alarms but would delay detection.

The synthetic network that we experimented with is shown in Figure 1; it has 100 nodes. It appears to have 4 autonomous systems (AS). The gateways are high degree nodes (important nodes). Gateways for AS 1 are nodes 32 and 21. Gateways for AS 2 are nodes 24, 67. Gateways for AS 3 are nodes 22, 44. Gateways for AS 4 are nodes 71, 33, 23. The node degree distributions, in this network, for the core cluster and for the "AS" networks are heavy tailed. The average path length of this network is 4.3.

We provide next a brief description of some of the experimental details in our investigations. Each node can act as a source or a sink. The flows of the network consist of "requests" in analogy to the "TCP-SYN" packets required for initiating a connection. The capacity of the links is infinite. The cost (in time) for a "request" to go through each link is of one time-step.

Under *normal mode* each node has a waiting time for initiating connections, which is modeled as either a Pareto or Exponential distribution. In either case the mean was set to two times the average path length in the network (round trip time (RTT)). A "Panic mode" simulates the situation where ¼ of the nodes in the network will attempt to contact at times t = 50 + G,..., t = 50 + 5 + G (where G is a random variable with Gamma distribution and parameters g = 2, t = 2) one of the nodes that are being monitored (node #6). Under an *anomaly mode*, an infection is modeled similar to the spread of Code Red I over the Internet, (we are considering vulnerable nodes). The attack starts with a single node infected at time t = 150. Once infected, a node spreads the self-propagating code by launching a specified number of threads, which generate random node numbers and then tries to contact them. Each thread waits for the round trip time after sending a request to the target before trying to contact any other node.

In figures 2- 4 below we present some representative samples of our investigations to date.

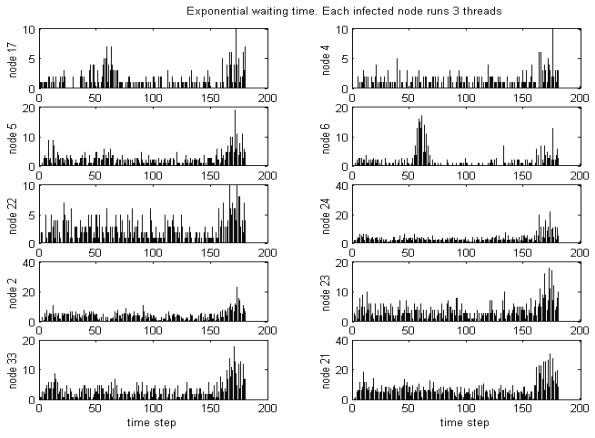

Figure 2: Flows when each infected node runs 3 threads

In the experimental network of Figure 1 we monitor the number of "TCP-SYN" packets going through 10 nodes with the highest degree (number of links). Nodes

21,22,23,24 and 33 are Gateways. The other nodes belong to the highly connected cluster in between the AS's. The surge of traffic flows at times immediately after the initiation of the spreading attack are quite obvious.

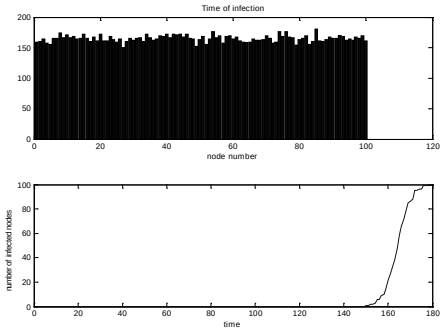

Figure 3: Infection time per node when the self-propagating code runs three threads

The basic problem formulation for automated change detection algorithms is as follows: Given the observed data from one or more nodes decide as fast as possible if a spreading attack is taking place. A more complicated problem is the combined detection/classification problem: Given several possible models or hypotheses that may have generated the observed data at one or more nodes decide as fast as possible if a spreading attack is taking place and the type of the attack. The methodologies we are using to analyze these problems proceed along two main ideas: developing generalized likelihood ratio (GLR) approach for on-line algorithms; developing filter bank algorithms (using HMMs). We are also investigating the development of robust non-parametric algorithms using cumulative sum (CUSUM) and Girshik-Rubin-Shiryaev (GRSh) statistics. In sequential versions of the problem the sequential probability ratio test (SPRT) is used.

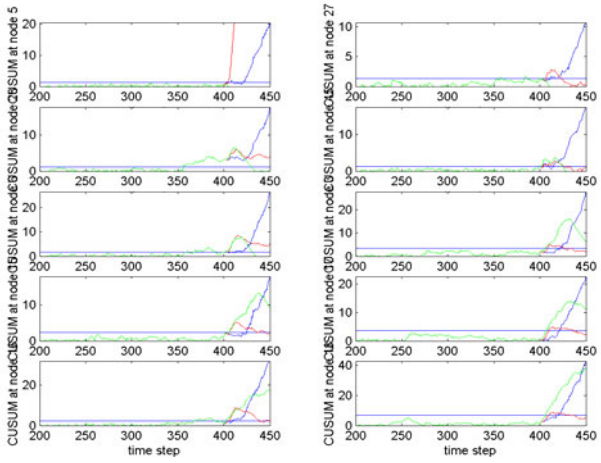

Figure 4: Preliminary picture showing the detection of the worm early in its development.

Change detection and quickest detection is a subject that has been investigated intensively through the years due to its wide applicability. We refer to the excellent references (Basseville and Nikiforov, 1993) (Shiryaev, 1978). Most of the algorithms and work to date have used i.i.d. observations. Performance of such an algorithm is shown in Figure 4 below.

3. CHANGE DETECTION IN A NETWORK FLOW PATTERN

3.1 Problem formulation

Most change detection algorithms applied to network traffic use non-parametric statistics as it is very complicated to know, or model the pre-change P_0 and the post-change P_1 distributions of an observation of the network flow at all times by parametric families (Blâzek et al., 2001;Wang et al. 2002.)

One such non-parametric sequential method to detect changes in the mean that we will be looking at is given by a threshold of the statistic:

$$S_k = \max\{0, S_{k-1} + N_k - m_k - c_k\} \quad (1)$$

where N_k can represent for example the number of packets seen in an interval of time $\Delta t = t/k$, m_k is a historical estimate of $E_0[N_k]$ and c_k is a positive deterministic sequence chosen experimentally to minimize the average detection delay. The stopping time is then

$$\tau = \min\{k : S_k \geq h\} \quad (2)$$

where *h* is the given threshold.

We take a new approach for identifying Distributed Denial of Service (DDoS) attacks by a set of nodes in a transit network. The basic idea is that at each highly connected node, the data tends to aggregate from the distributed sources towards the destination, giving a sense of "directionality" to the attack. This directionality concept provides us a framework to design change detection algorithms that are going to be less sensitive to changes in the average intensity of the overall traffic and will focus in differentiating the different random fluctuations of the network traffic versus fluctuations where there is a clear change in the direction of the flow at a given node. We are considering "packets" in a very broad and general way, but clearly our approach can be extended to monitor certain specific packet types given the right protocol. For example we might be interested in measuring only TCP SYN-ACK response packets for identifying a "reflected distributed denial of service attack", or ICMP packets for identifying ping floods.

Let's assume we are monitoring node *d* in Figure 5. Let $X_k^{d,m}$ denote the total number of packets sent by *d* through the link *(d,m)*, where $m \in N(d)$ denotes a neighbor of *d*, N(d). Let X_k^d denote the vector with elements $X_k^{d,m}$ and let

$$\theta_0^d(k) := \begin{bmatrix} E_0[X_k^{d,a}] \\ E_0[X_k^{d,b}] \\ E_0[X_k^{d,c}] \end{bmatrix}$$

We are interested in changes of the form:

$$\theta_0^d(k) + v\Upsilon := \begin{bmatrix} E_0[X_k^{d,a}] \\ E_0[X_k^{d,b}] \\ E_0[X_k^{d,c}] \end{bmatrix} + v\Upsilon \qquad (3)$$

where v is a non-negative scalar and Υ (in the case of three observed links) is one of the usual basis vectors of the three dimensional Euclidean space. Namely:

$$\Upsilon_a = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \Upsilon_b = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \Upsilon_c = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

So in Figure 5, if node *d* suddenly starts a broadcast, there will be a change in the mean of all the processes, but we are not interested in such a change. Instead, if there are attackers in the sub networks attached to *b* and *c*, and they target a host in the network attached to *a* by flooding it, there will be a change in the direction Υ_a. Testing "directions" should help us discriminate unwanted false alarms due random fluctuations of flows.

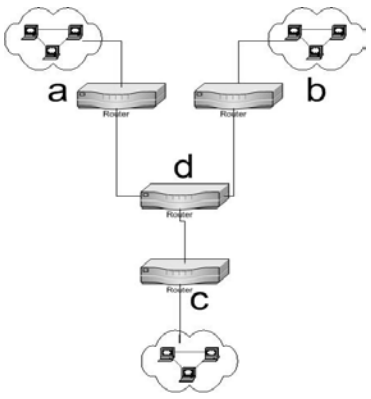

Figure 5: Transit network composed of nodes a, b, c, d. We monitor all outgoing links of node d.

To formalize our ideas we consider the framework discussed in (Basseville and Nikiforov, 1993) of change detection in a known direction but unknown magnitude of the change. Our problem is a little bit different in that we are considering an M-ary sequential hypothesis testing problem and in that we not allow changes with negative values for v, i.e. we impose the restriction $v \geq 0$

Thus the resulting change detection problem is:

$$\theta^d(k) = \begin{cases} \theta_0^d & when \quad k < t_{change} \\ \theta_0^d + v\Upsilon_a & or \\ \theta_0^d + v\Upsilon_b & or \\ \theta_0^d + v\Upsilon_c & when \quad k \geq t_{change} \end{cases} \qquad (4)$$

where t_{change} is an unknown time when the change occurs. We run in parallel a GLR in each possible direction Υ_m (*m* a neighbor of *d*) *vs* the null hypothesis assuming θ_0^d:

$$g_k^{d,m} = \max_{1 \leq j \leq k} \ln \frac{\sup_{v \geq 0} \prod_{i=j}^{k} p_{\theta_0^d + v\Upsilon_m}(X_k^d)}{\prod_{i=j}^{k} p_{\theta_0^d}(X_k^d)} \qquad (5)$$

Only the test *m* that reaches its given threshold is stopped. The threshold $h^{d,m}$ for each of the parallel tests is selected given a fixed false alarm rate probability.

Equation (5) has a closed form solution when the distributions are assumed to be Gaussian $\mathcal{N}(\theta_0^d, \Sigma_d)$. In this case we get the constrained optimization problem:

$$\min_{v \geq 0} \sum_{i=j}^{k} \frac{1}{2} \left(X_i^d - \theta_0^d - v\Upsilon_m\right)^T \Sigma_d^{-1} \left(X_i^d - \theta_0^d - v\Upsilon_m\right) \qquad (6)$$

If the restriction is inactive ($\lambda = 0$), then we obtain

$$\hat{v}_k^{d,m}(j) = \frac{\Upsilon_m^T \Sigma_d^{-1} \left(\frac{1}{k-j+1} \sum_{i=j}^{k} X_i^d - \theta_0^d \right)}{\Upsilon_m^T \Sigma_d^{-1} \Upsilon_m} \qquad (7)$$

If the restriction is active, then $\hat{v}_k^m(j) = 0$

Intuitively we are projecting the difference between all available sample means and the mean θ_0^d into each of the possible directions Υ_m, selecting the time step that maximizes the likelihood of the alternate hypothesis assuming $\theta_0^d + \hat{v}_k^{d,m}(j)\Upsilon_m$, i.e. when we "think" the sample mean started moving in the Υ_m direction. The uncorrelated covariance (in our case) is just a weighting parameter for being cautious about declaring a change in the mean too soon if the process is observed to have large fluctuations around the mean.

We tested the robustness of this approach even when the distribution is not Gaussian, as long as we are computing the mean and covariance in a window of time, much larger than our false alarm delay, to keep mean and covariance "up to date". Although the number of packets through a link does not follow a Gaussian distribution, the analysis is still valid based on second order statistics of the flows. The mean and the variance are easily learned and the model is used to provide an approximate picture

of changes in direction of the flow. We are primarily concerned here about changes in the direction of the flow and not its intensity per se.

We also use the non-parametric test $S_k^{d,m}$ given by equation (1) to test for changes in the mean of the utilization of the link *(d,m)* in the time interval *k*. Note that one main difference is that for testing a change in the link *m*, $S_k^{d,m}$ does not use information from the other links of node *d* whereas $g_k^{d,m}$ does. A possible use of the information contained in all the links from *d* is to compute a non-parametric statistic that measures the changes in the normalized traffic seen through a link *(d,m)*:

$$\rho_k^{d,m} := \frac{X_k^{d,m}}{\sum_{m \in N(d)} X_k^{d,m}} \quad (8)$$

This approach has the added advantage that a positive change in the mean of $\rho_k^{d,m}$ tends to yield a negative change in the mean on any other neighbor *n* of *d* as $\rho_k^{d,m}$ and $\rho_k^{d,n}$ are negatively correlated because:

$$\sum_{m \in N(d)} \rho_k^{d,m} = 1$$

Experimental validation shows that the process $\rho_k^{d,m}$ has fewer variations than its unormalized counterpart and will be more amenable to a mean computation and the usage of the non-parametric statistic (1).

3.2 Correlation mechanism

So far we have been focusing on detecting a change in a single node. One of the main advantages in having several nodes under monitoring is that we can perform a correlation of the statistics between the different nodes in order to decrease the detection delay given a fixed false alarm rate probability. The alarm correlation can be performed by several methods. Here we propose a simple algorithm that will only require the knowledge of the routing tables for the nodes being monitored.

We want a mechanism to aggregate the different statistics at each monitored node. Clearly the correlation mechanism cannot be multiplicative, because if we are monitoring a node physically unable to detect the attack (a node that is not in the routing path of any of the attackers and the victim) the low value of the computed statistic of this node will adversely affect any small information that any other node might have related to the attack. On the other hand the computed statistics g_k for all nodes can vary to different scales of magnitude yielding a biased addition. To cope with this problem we compute the normalized statistic

$$\varphi_k^{d,m} := \frac{g_k^{d,m}}{h^{d,m}} \quad (9)$$

If none of our monitored nodes has raised an alarm, the number of monitored nodes will bound $\sum_d \varphi_k^{d,m}$. This can be in turn interpreted as a new upper bound for a "collective" threshold and can be selected given a false alarm rate probability. Selecting which statistics to correlate (add) is a key issue. In keeping with our "directionality" framework we will correlate only the statistics relating two or more nodes to a common node. That is the reason we need the routing table information of our monitored nodes.

The algorithm is as follows:

```
Given two nodes d and e,
For each link d->m in d{
      For each link e->n in e{
            If there is a node f
            reachable through the
            routing tables of d->m and
            e->n, then correlate the
            normalized statistic of
            d->m with the statistic of
            e->n
      }
}
```

In the following section we apply this formulation for the case of two nodes, but it can be extended recursively when we are monitoring three or more nodes.

4. SIMULATION RESULTS AND EVALUATION

For our experimental results we used the network simulation software ns2. We created a script to generate a random scale-free transit network topology with a given number of sub networks. We will focus in one of our realizations given in Figure 6. It consists of 15 transit nodes performing only routing between 12 subnetworks, each with 65 hosts each. During the normal operation of the network each of these 780 hosts selects randomly a host in another sub network, and establishes an On-Off source connection with Pareto distributed times. The routing protocol selects a route with the least number of hops towards a given destination.

The attack is simulated with a given number of compromised nodes in different sub networks. During the attack, each of these nodes will start a constant bit rate connection towards a specific node. The rate of the attackers was varied to test the detection algorithm with different percentage of attack packets circulating over the transit network at a given time. We considered 7 attackers. One in each of the sub networks connected to nodes 3, 4, 5, 8, 9, 11 and 13. The victim is in the network connected to node 14.

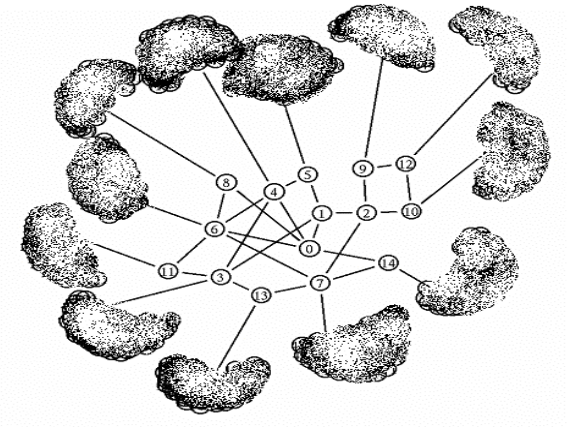

Figure 6: The transit network consists of 15 routers Each "cloud" represents a subnetwork

Some typical link usage characteristics can be seen in Figure 7. We first tested the performance of the statistics $S_k^{d,m}$ and $g_k^{d,m}$ at individual nodes. With the network under normal operation we experimentally obtained the thresholds for each statistic for a false alarm rate of 0.003.

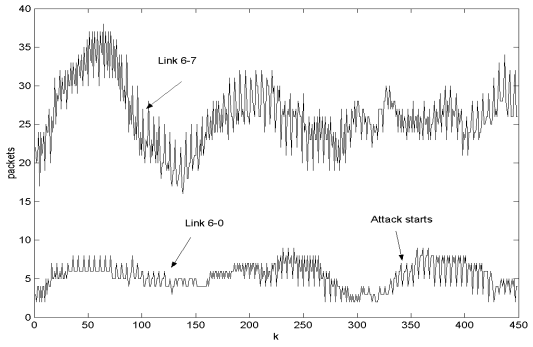

Figure 7: Node 6 uses node 0 to reach 14. An attack occupying 2.5% percent of the transit network traffic was started at k = 350.

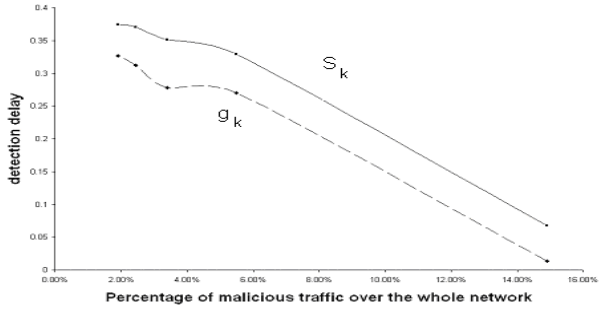

Figure 8: Average detection delay per node when we monitor nodes 0, 1, 2, 7. Detection delay of 1 is same as the average delay for a false alarm.

We selected the nodes 0, 1, 2, 7 to test the detection delay of the statistics independently of each other. The results can be seen in Figure 8, where the average delay of detection is computed between the four nodes for different percentages of the amount of traffic the attack generates over the transit network. Node 0 had the smallest delay, as it is a node where most of the traffic towards 14 gets agglomerated, giving a large change in the mean. $g_k^{d,m}$ performs marginally better than $S_k^{d,m}$.

The correlation mechanism can be applied to decrease the detection delay when we are monitoring more than one node. Lets consider the case of two nodes far (in the number of hops) from the victim. In Figure 6 we will pick nodes 6 and 3 for the correlation. Furthermore if we consider an attack reaching less than 2% of the transit network traffic it will be very difficult to detect any abnormal change without a global integrated view of the statistics at different nodes. For testing our correlation algorithm we will start at k = 1 an attack reaching 1.5% of all traffic at the transit network. Again the attack is towards a host in the sub network of node 14.

The normalized statistics are computed for each link in the nodes. Figure 9 shows the normalized statistics for node 6. The routing tables required for the correlation algorithm are in Tables 1, 2.

TABLE 1: ROUTING TABLE FOR NODE 6

Link	Routing to nodes:
(6,7)	7,13,2,10,12,9
(6,0)	0,14,1
(6,4)	4,3,5
(6,11)	11,3
(6,8)	8
(6,sub network)	

TABLE 2: ROUTING TABLE FOR NODE 3

Link	Routing to nodes:
(3,1)	1,0,2,14,10,9,12
(3,13)	7,13
(3,4)	4,5
(3,11)	11,6,8
(3,sub network)	

By simple inspection of the routing tables we see that we need to correlate link (6,0) with (3,1) because nodes 6 and 3 use them to reach nodes 0, 1 and 14. Similarly, the link (6,11) must be correlated with (3,11), link (6,4) with (3,4), link (6,7) with (3,13), and (6,7) with (3,1).

The results of our correlation between all allowable normalized statistics are shown in Figure 10. It is clear that the correlation of the normalized statistics at nodes 6 and 3 gives a better resolution of the attack than the statistics of 6 and 3 alone while discriminating the

uncorrelated random fluctuations of the traffic intensity that cause most of the false alarms.

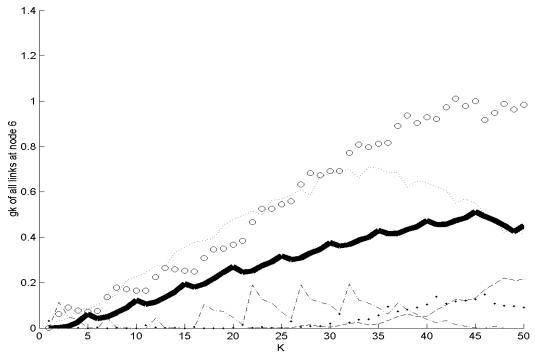

Figure 9: All normalized statistics for node 6. Solid dark curve is the normalized statistic for link (6,0). Circles identify the statistic from 6 to its sub net. This statistic raises a false alarm at k=42.

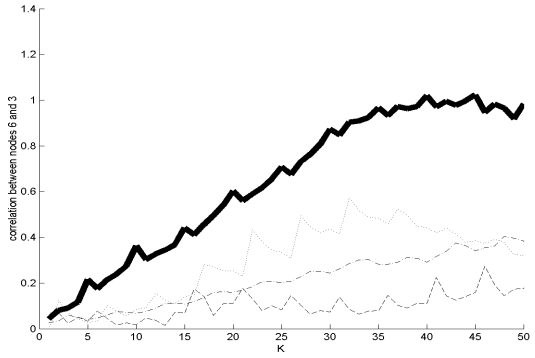

Figure 10: Dark solid curve is the correlated statistic of links (6,0) and (3,1). The other dotted curves represent the correlated statistics of the remaining allowable links.

Not only can we detect the attack (depending on the new correlation threshold), but also we can diminish the impact of the false alarm originating at node 6. However another important conclusion is that without the need to extract or store header information from the packets transmitted through the network, we are able to infer (from the intersection of the two routing tables for the "winning" correlated statistic of the links (6,0) and (3,1)) the only three possible targets: Namely nodes 0, 1 and 14.

A simple marginal constant reduction in the delay detection such as that obtained with $g_k^{d,m}$ vs. $S_k^{d,m}$ can provide significant help when we correlate the statistics because constant gains will get multiplied by the number of nodes participating in the detection reducing exponentially the detection delay.

CONCLUSIONS

In this paper we have investigated the problem of detecting anomalous behavior and distributed attacks in a network. We investigated detection of spreading of active code based on the spatio-temporal pattern variations in the flows of a set of nodes. We also investigated detection of distributed denial of service attacks. We have formulated these problems as distributed change detection problems on a graph. We described several algorithms and their performance.

Future work includes: identification of the most promising models to use for the network and for the data generation mechanisms for the monitoring data; learning either the parameters of the models assumed for normal or abnormal data generation, or learning the corresponding data pattern; more detailed analysis of the distributed "quickest detection" on graphs; analysis of the trade-off between correct detection and/or classification and false alarms; investigation of these problems in the context of mobile wireless networks.

ACKNOWLEDGMENTS

This material is based upon work supported by the U.S. Army Research Office under Award No. DAAD19-01-1-0494 to the University of Maryland College Park.

REFERENCES

Albert, R. and Barabasi A.-L. "Statistical Mechanics of Complex Networks," Reviews of Modern Physics,pp. 47-97, January 2002.

Basseville, M. and Nikiforov I.V. *Detection of Abrupt Changes: Theory and Application*, Englewood Cliffs, NJ: Prentice Hall, 1993.

Blâzek, R.B., Kim H. Rozovskii B. and Tartakovsky A. "A novel approach to detection of denial-of-service attacks via adaptive sequential and batch-sequential change-point detection methods," IEEE Systems, Man and Cybernetics Information Assurance Workshop, June 2001.

Bohacek, S., "Optimal Filtering for Denial of Service Mitigation," IEEE Conf. on Dec. and Control, 2002.

Houle K.J and Weaver, G.M. "Trends in Denial of Service Attack Technology" CERT Coordination Center v1.0 October 2001.

Shiryaev, A.N., *Optimal Stopping Rules*, Springer, 1978.

Staniford S., Paxson V. and Weaver N. "How to 0wn the Internet in your Spare Time," Proceedings of the 11th USENIX Security Symposium (Security '02) 2002.

Wang, H. Zhang D. and Shin K.G. "Detecting SYN Flooding Attacks," Proceedings of INFOCOM 2002, New York City, New York, June,2002.